FIG. 1

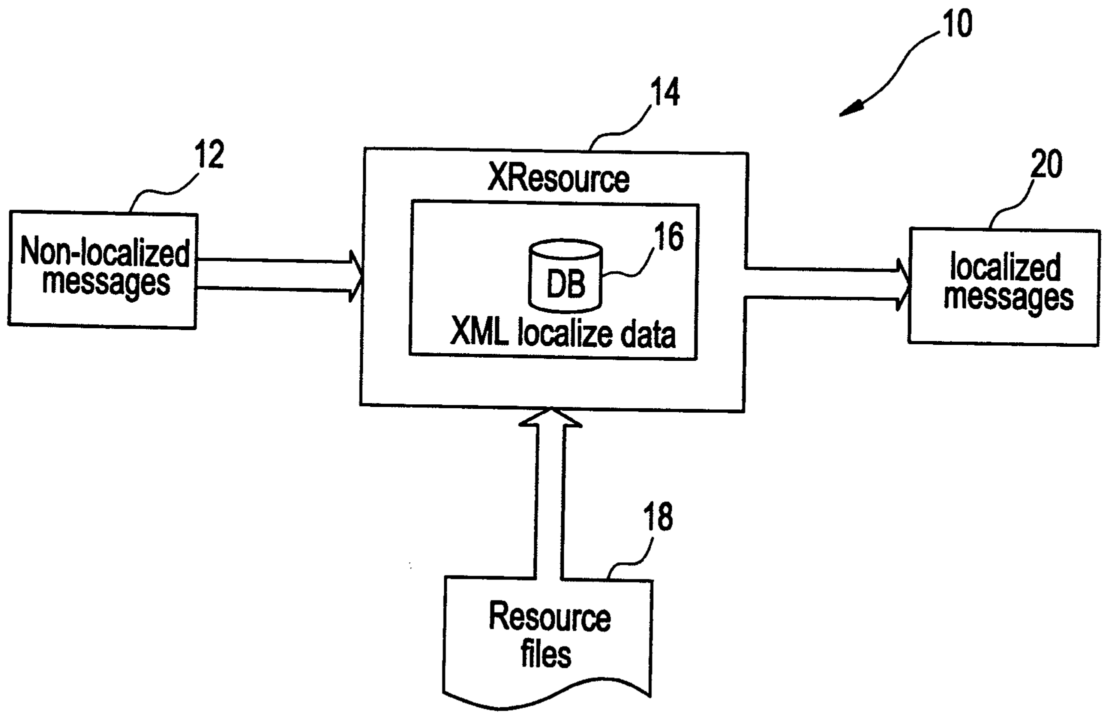

10
12
Non-localized messages
14
XResource
16
DB
XML localize data
18
Resource files
20
localized messages

FIG. 2A

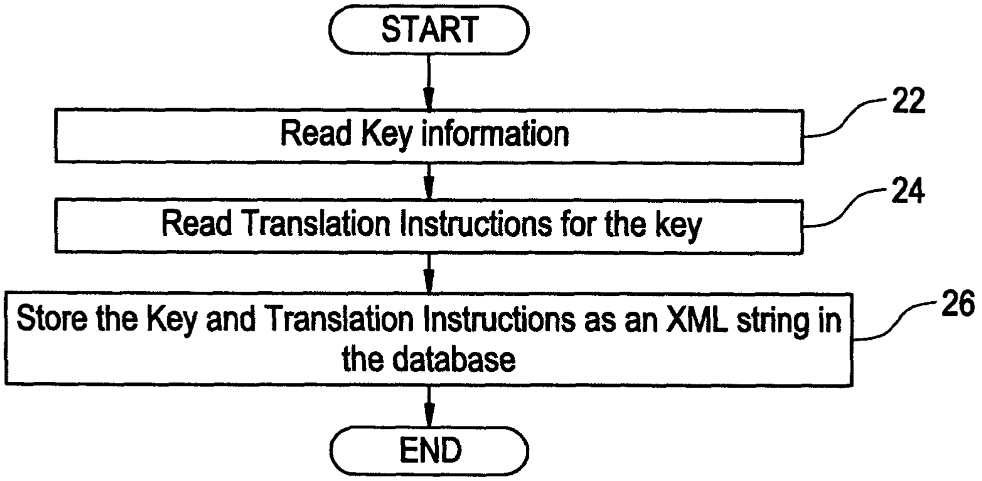

START
Read Key information
22
Read Translation Instructions for the key
24
Store the Key and Translation Instructions as an XML string in the database
26
END

FIG. 2B

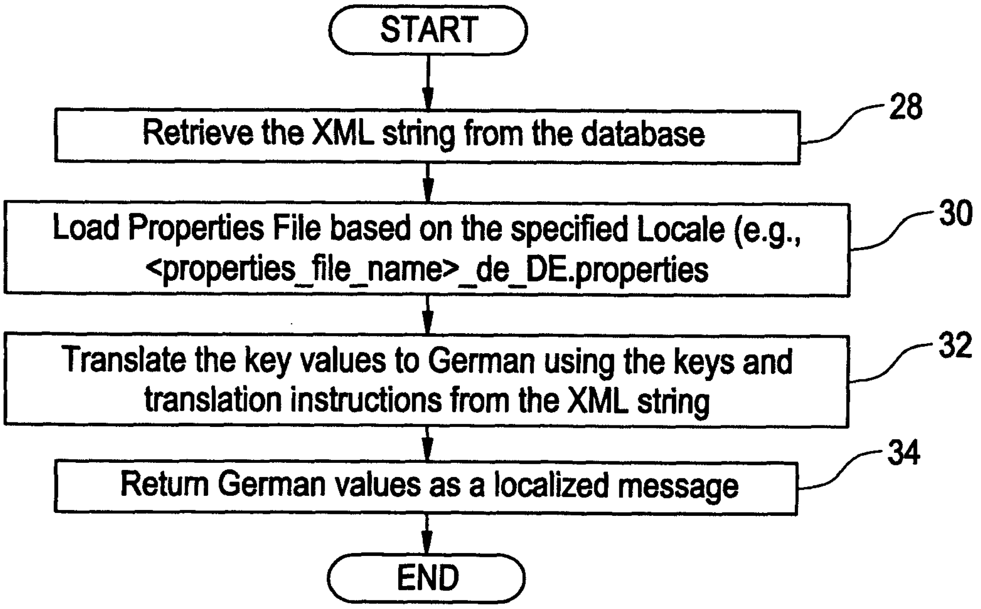
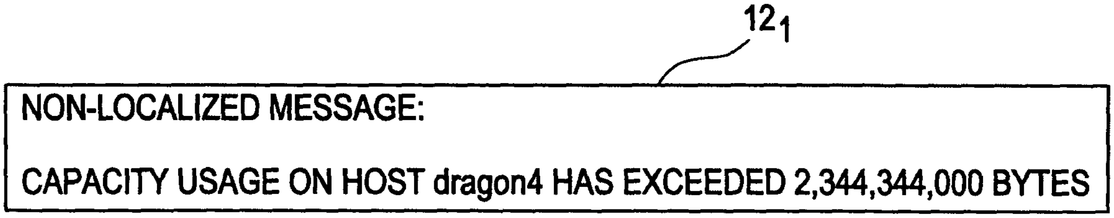

START
Retrieve the XML string from the database
28
Load Properties File based on the specified Locale (e.g., <properties_file_name>_de_DE.properties
30
Translate the key values to German using the keys and translation instructions from the XML string
32
Return German values as a localized message
34
END

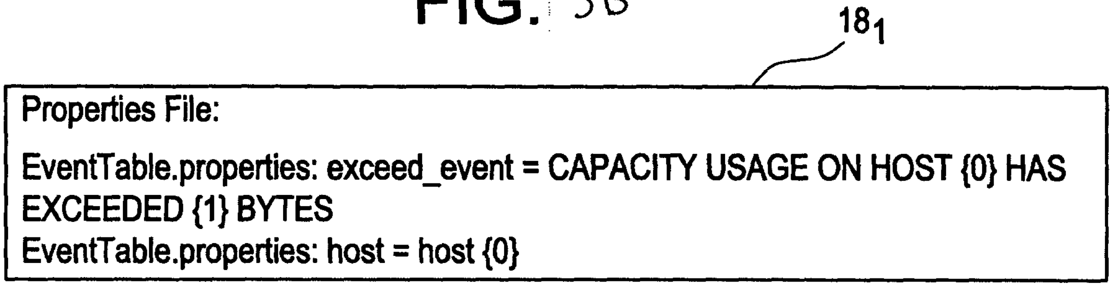
FIG. 3A



NON-LOCALIZED MESSAGE:

CAPACITY USAGE ON HOST dragon4 HAS EXCEEDED 2,344,344,000 BYTES

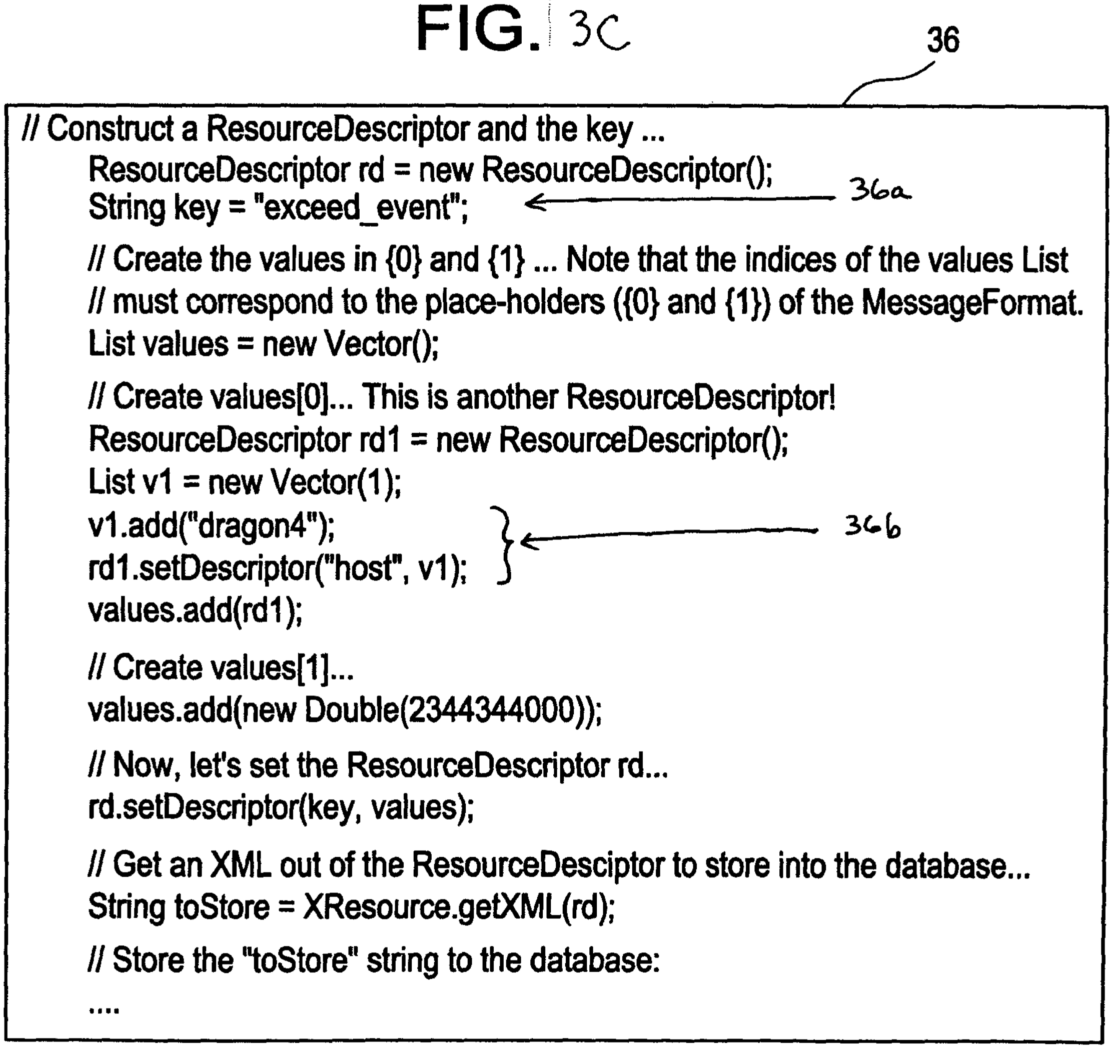
FIG. 3B



Properties File:

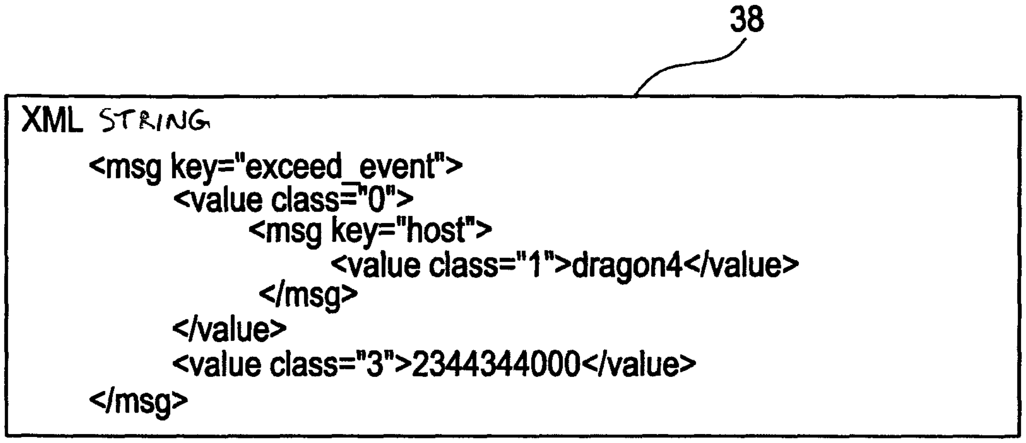
EventTable.properties: exceed_event = CAPACITY USAGE ON HOST {0} HAS EXCEEDED {1} BYTES
EventTable.properties: host = host {0}

FIG. 3C



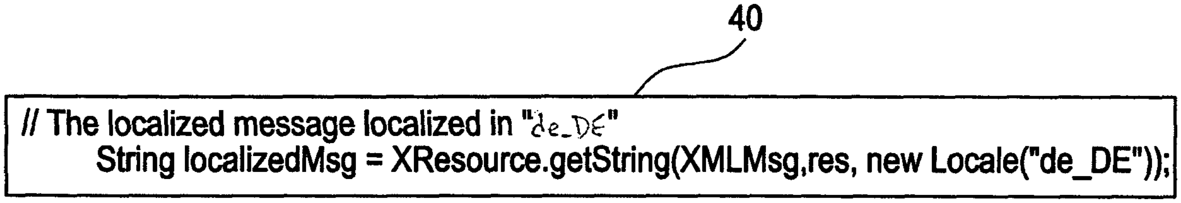
```
// Construct a ResourceDescriptor and the key ...
      ResourceDescriptor rd = new ResourceDescriptor();
      String key = "exceed_event";

      // Create the values in {0} and {1} ... Note that the indices of the values List
      // must correspond to the place-holders ({0} and {1}) of the MessageFormat.
      List values = new Vector();

      // Create values[0]... This is another ResourceDescriptor!
      ResourceDescriptor rd1 = new ResourceDescriptor();
      List v1 = new Vector(1);
      v1.add("dragon4");
      rd1.setDescriptor("host", v1);
      values.add(rd1);

      // Create values[1]...
      values.add(new Double(2344344000));

      // Now, let's set the ResourceDescriptor rd...
      rd.setDescriptor(key, values);

      // Get an XML out of the ResourceDesciptor to store into the database...
      String toStore = XResource.getXML(rd);

      // Store the "toStore" string to the database:
      ....
```

FIG. 3D



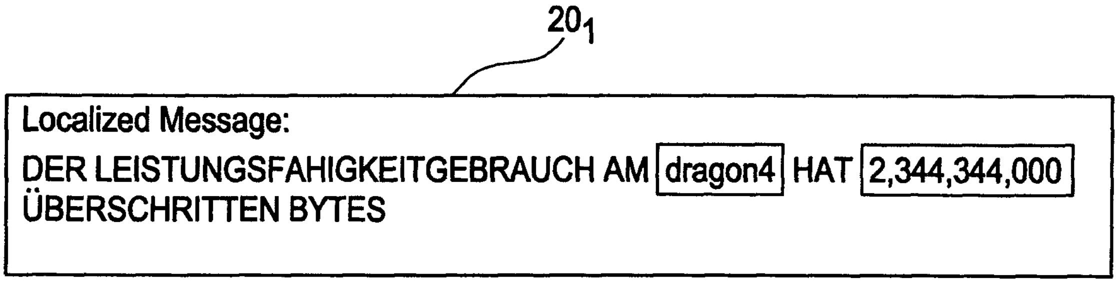
```
XML STRING
      <msg key="exceed_event">
              <value class="0">
                     <msg key="host">
                             <value class="1">dragon4</value>
                      </msg>
              </value>
              <value class="3">2344344000</value>
      </msg>
```

FIG. 3E



```
// The localized message localized in "de_DE"
      String localizedMsg = XResource.getString(XMLMsg,res, new Locale("de_DE"));
```

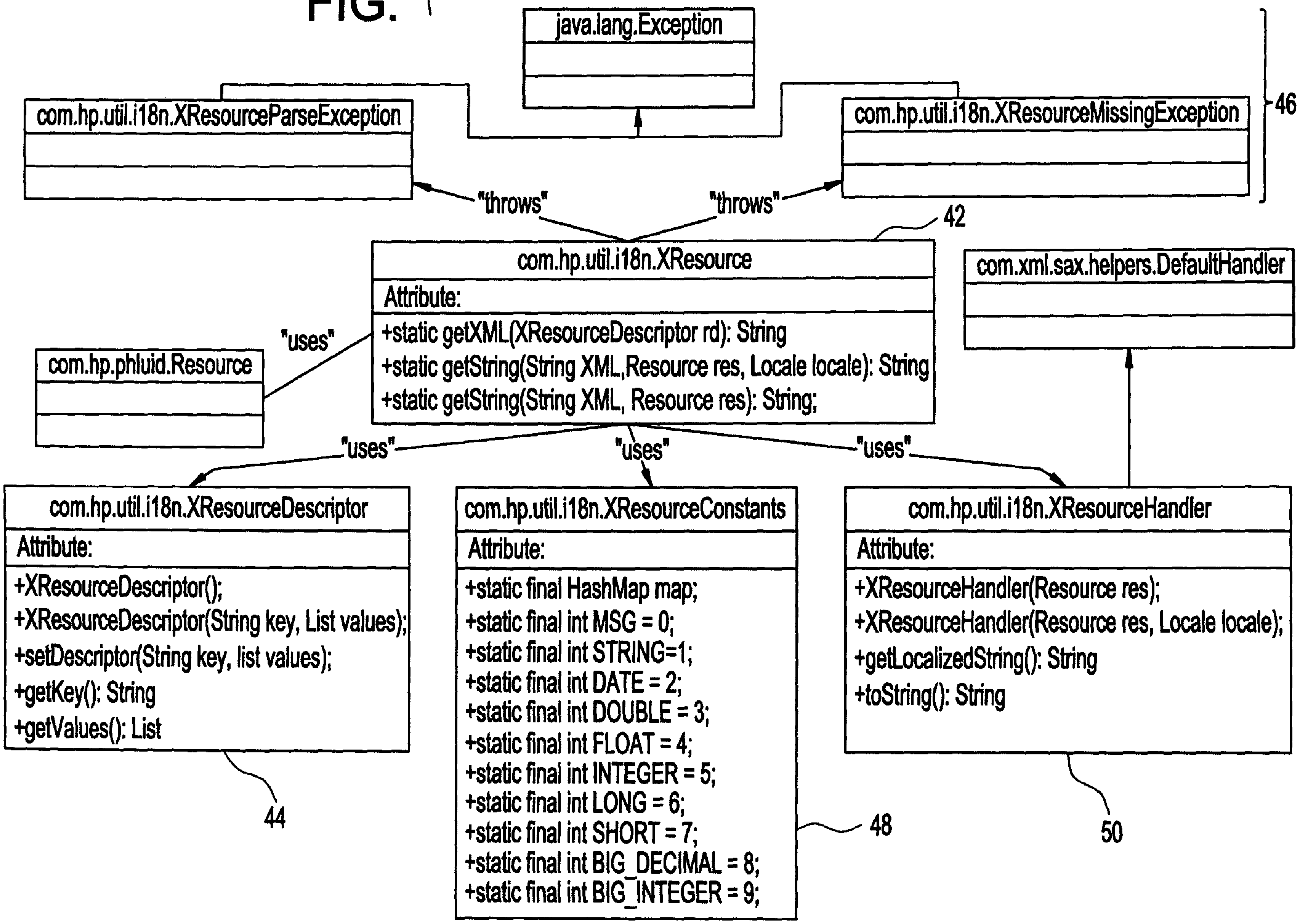
FIG. 3F



Localized Message:

DER LEISTUNGSFAHIGKEITGEBRAUCH AM dragon4 HAT 2,344,344,000 ÜBERSCHRITTEN BYTES

FIG. 4


java.lang.Exception
com.hp.util.i18n.XResourceParseException
com.hp.util.i18n.XResourceMissingException
46
"throws"
"throws"
42
com.hp.util.i18n.XResource
Attribute:
+static getXML(XResourceDescriptor rd): String
+static getString(String XML,Resource res, Locale locale): String
+static getString(String XML, Resource res): String;
com.xml.sax.helpers.DefaultHandler
"uses"
com.hp.phluid.Resource
"uses"
"uses"
"uses"
com.hp.util.i18n.XResourceDescriptor
Attribute:
+XResourceDescriptor();
+XResourceDescriptor(String key, List values);
+setDescriptor(String key, list values);
+getKey(): String
+getValues(): List
com.hp.util.i18n.XResourceConstants
Attribute:
+static final HashMap map;
+static final int MSG = 0;
+static final int STRING=1;
+static final int DATE = 2;
+static final int DOUBLE = 3;
+static final int FLOAT = 4;
+static final int INTEGER = 5;
+static final int LONG = 6;
+static final int SHORT = 7;
+static final int BIG_DECIMAL = 8;
+static final int BIG_INTEGER = 9;
com.hp.util.i18n.XResourceHandler
Attribute:
+XResourceHandler(Resource res);
+XResourceHandler(Resource res, Locale locale);
+getLocalizedString(): String
+toString(): String
44
48
50